REMARKS

Applicants are in receipt of the Office Action mailed October 4, 2004. Claims 1-28 are pending in the application. Reconsideration is respectfully requested in light of the following remarks.

Double Patenting Rejection:

The Office Action rejected claims 1-28 under the judiciary created doctrine of obviousness-type double patenting as being unpatentable over claims 1-2, 8-23 and 29-43 of U.S. Patent No. 6,792,466. Applicants traverse this rejection. However, to expedite prosecution of the application, a terminal disclaimer to obviate the double patenting rejection over claims 1-28 has been filed along with this response. Accordingly, applicants respectfully request removal of the double patenting rejection of claims 1-28.

Section 103(a) Rejection:

The Office Action rejected claims 1-7, 14-20 and 27 under 35 U.S.C. § 103(a) as being unpatentable over Juster et al. (U.S. Patent 6,202,089) (hereinafter "Juster") in view of Stern et al. (U.S. Patent 5,935,249) (hereinafter "Stern").

Regarding claim 1, contrary to the Examiner's assertion, Juster in view of Stern fails to teach receiving an address for a service within the distributed computing environment; linking the address to a pre-generated message interface for accessing the service, wherein the message interface comprises computer-executable code built into the device, and wherein the linking creates a message endpoint for the device to send messages to the service at the address in order to access the service; and using the message endpoint to send messages to the address to access the service.

Juster teaches a method for configuring, identifying and using a plurality of remote procedure call (RPC) endpoints in a single server process. Specifically, Juster

teaches a server process that establishes a plurality of RPC endpoints and a separate RPC endpoint for responding to address queries from clients. According to Juster, a client first places a remote procedure call to the address request RPC endpoint of the server process that returns the address of another RPC endpoint providing a service desired by the client (Juster, Abstract, column 2, lines 3-26).

Contrary to the Examiner's contention, Juster in view of Stern fails to teach receiving an address for a service within the distributed computing environment and <u>linking the address to a pre-generated message interface for accessing the service</u>. The Examiner cites the Abstract, Figures 2A, 2B, and column 2, lines 3-26 and column 7-lines 10-31 of Juster. However, none of the cited portions of Juster disclose *linking a received address to a pre-generated message interface* for accessing the service. Instead, Juster teaches that a client queries the server via the server's address request RPC endpoint to obtain the address of a RPC endpoint that provides a desired service (Juster, column 2, lines 18-23, column 4, lines 55-52, and column 7, lines 10-31). Once the client obtains the address, Juster states only that the client performs remote procedure calls on the server RPC endpoint using the received address (Juster, Figure 5, step 525, column 2, lines 23-25, and column 4, lines 62-64). Juster does not mention a client linking the address received from the server to a pre-generated message interface for accessing the service. In fact, Juster is completely silent regarding how a client performs remote procedure calls because Juster is not concerned with how a client performs remote procedure calls. Instead, Juster deals only with configuring multiple RPC endpoints for a single server process. Thus, Juster in view of Stern fails to teach or suggest receiving an address for a service within a distributed computing environment and linking the address to a pre-generated message interface for accessing the service.

Juster in view of Stern further fails to teach linking an address to a pre-generated message interface for accessing the service, wherein said linking <u>creates a message endpoint</u> for the device to send messages to the service at the address in order to access the service, contrary to the Examiner's assertion. As noted above, Juster fails to disclose anything regarding how a client makes remote procedure calls after obtaining an address

for one of the server's RPC endpoints. In fact, Juster only states, "the client then places a remote procedure call on the desired endpoint of the server process," without providing any further information regarding how a client may use one of the server's RPC endpoint addresses to make a remote procedure call (Juster, column 2, lines 23-26).

None of the Examiner's cited portions of Juster (Abstract, Figures 2A, 2B, and column 2, lines 3-26 and column 7-lines 10-31) mention anything regarding linking of an address creates a message endpoint for the device to send messages to the service at the address in order to access the service. Instead, the portions cited by the Examiner only provide an overview of Juster's invention, which, as noted above, deals with configuring multiple RPC endpoints for a server process, but does not disclose anything about creating a message endpoint for the device to send messages to the service at the address in order to access the service (Abstract, Figures 2A and 2B, and column 2, lines 3-26). The Examiner also cited a passage where Juster describes how a service responds to a client address query by determining an appropriate RPC endpoint and providing an address to that RPC endpoint to the requesting client (column 7, lines 10-31). However, as noted above, Juster does not describe, or even mention, how a client uses the provided address except to say that the client performs remote procedure calls on the server RPC endpoint using the received address (Juster, Figure 5, step 525, column 2, lines 23-25, and column 4, lines 62-64). Juster mentions nothing of linking a received address to a pre-generated message interface. Additionally, Stern does not overcome any of the deficiencies of Juster regarding linking an address to a pre-generated message interface creates a message endpoint for the device to send messages to the service at the address in order to access the service.

In further regard to claim 1, Juster in view of Stern also fails to teach linking said address to a pre-generated message interface for accessing said service, wherein the <u>message interface comprises computer-executable code built into the device</u>. The Examiner admits that Juster fails to teach a pre-generated message interface comprising computer-executable code built into the device and relies upon Stern, citing the Title, Figures 4, 6, 7, and 9, as well as column 4, lines 1-14 and column 8, lines 27-59 of Stern.

However, none of the cited portions of Stern disclose a pre-generated message interface comprising computer-executable code built into the device. Instead, the Title only refers to a mechanism for embedding network based control systems in a local network interface device. Figures 4, 6, 7, and 9 illustrate various networked computer devices, none of which include a pre-generated message interface comprising computer-executable code built into the device. Column 4, lines 1-14 of Stern only describe an exemplary computer system that may load and execute instructions, either from a persistent store, or downloaded over the network. Column 8, lines 27-59 of Stern describe Stern's use of Java and a Java Virtual Machine in Stern's Java Enabled Network Interface Device. Thus, none of the Examiner's cited portions of Stern disclose or mention anything regarding a pre-generated message interface comprising computer-executable code built into the device.

Traditional remote procedure call (RPC) systems, such as those mentioned by both Juster and Stern, utilize one of two different client-side RPC mechanisms. In some traditional RPC systems, a client wishing to communicate with a remote server locates and downloads a stub object that is configured to communicate with the remote server object (or process). The client then executes methods of the stub object that communicates the necessary method parameters and return values between the client and the remote object. In other traditional RPC systems, the client repeatedly calls a standard RPC library routine, passing in the remote RPC endpoint address and any necessary parameters. The client calls the same routine to execute RPC calls on different remote server objects, passing in the relevant address each time. Conventional RPC mechanisms, such as described in Juster and Stern do not *link an address to a pre-generated message* interface for accessing a service, wherein the message interface comprises computer-executable code *built in to the device.*

The combination of Juster and Stern suggested by the Examiner would only result in a system including a server process that establishes multiple RPC endpoints, including an address query endpoint, and that also includes a Java Enabled Network Interface Device as taught by Stern. Since neither Juster nor Stern, alone or in combination, teach

or suggest linking an address to a pre-generated message interface for accessing the service, wherein the message interface comprises computer-executable code built in to the device, and wherein the linking creates a message endpoint for the device to send messages to the service at the address in order to access the service, the rejection is unsupported by the cited art.

Thus, in light of the above remarks, Applicants assert that the rejection of claim 1 is not supported by the cited prior art and its withdrawal is respectfully requested. Similar remarks as discussed above regarding claim 1, also apply to claims 14 and 27.

Regarding claim 2, contrary to the Examiner's assertion, Juster in view of Stern fails to teach the message interface of the message endpoint verifying that the messages sent to the service comply with a message schema for the service. The Examiner cites portions of Juster (Abstract, Figures 2A and 2B, and column 2, lines 3-26). However, none of the cited portions of Juster teach or disclose a message interface of a message endpoint verifying that messages sent to a service comply with a message schema for the service. Instead, as noted above regarding the rejection of claim 1, Juster only states that the client performs remote procedure calls on the server RPC endpoint using the received address (Juster, Figure 5, step 525, column 2, lines 23-25, and column 4, lines 62-64). Juster does not mention anything regarding a message schema for the service, nor about any sort of message interface verifying that messages sent to a service comply with a message schema for the service. Additionally, the Examiner has failed to point out or cite any portion of either Juster or Stern that describes a message interface verifying that messages sent to a service comply with a message schema for the service. As noted above regarding claim 1, Juster is not concerned with how a client access a service, except that a client first contacts a service via an address query RPC endpoint to obtain an address to a RPC endpoint providing a specific service. Juster does not teach, nor does Juster's system include, any sort of message interface verifying that messages sent to a service comply with a message schema for the service. As with claim 1, above, Stern fails to overcome any of Juster's deficiencies regarding the Examiner's rejection of claim 2.

Thus, for at least the reasons above, the rejection of claim 2 is not supported by the prior art and its removal is respectfully requested. Similar remarks as discussed above in regard to claim 2, apply to claim 15.

In regard to claim 3, contrary to the Examiner's assertion, Juster in view of Stern fails to teach wherein the message schema defines messages to be sent to and received from the service, wherein the messages are defined in a data representation language. The Examiner cites column 9, lines 3-47 of Stern. However, the portion of Stern cited by the Examiner does not describe any message schema defining messages to be sent to and received from a service and further does not teach that the messages are defined in a data representation language. Instead, column 9, lines 3-47 of Stern, describe how Stern's system allows for a Java Virtual Machine to securely store an object of value, such as a software license or other software key controlling the use of an application. The Examiner's cited portion of Stern also describes how his Java Enabled Network Interface may be treated as a trusted device for the purposes of reloading items of value, such as for resetting the number of uses permitted by a software license. A Java Enabled Network Interface securely storing security objects and Stern's other objects of value, does not disclose a message schema defining messages to be sent to and received from a service, nor does it teach that such messages may are defined in a data representation language. Stern does not teach, in the Examiner's cited passage or elsewhere, a message schema defining messages to be sent to and received from a service, and wherein the messages are defined in a data representation language.

Therefore, the Examiner's proposed combination of Juster in view of Stern also fails to teach wherein the message schema defines messages to be sent to and received from the service, wherein the messages are defined in a data representation language. Additionally, the arguments presented above regarding claim 2 apply to claim 3, as well. Thus, for at least the reasons above, the rejection of claim 3 is not supported by the prior art and its removal is respectfully requested. Similar remarks as discussed above in regard to claim 3, apply to claim 16.

Regarding claim 4, Juster in view of Stern fails to teach wherein the data representation language is eXtensible Markup Language, as asserted by the Examiner. The Examiner cites column 9, lines 3-47 of Stern. However, the portion of Stern cited by the Examiner does not describe any message schema defining messages to be sent to and received from a service, wherein the messages are defined in a data representation language; and wherein the data representation language is eXtensible Markup Language. Instead, column 9, lines 3-47 of Stern, describe how Stern's system allows for a Java Virtual Machine to provide secure storage of an object of value, such as a software license or other software key controlling the use of an application. The Examiner's cited portion of Stern also describes how a Java Virtual Machine may be treated as a trusted device for the purposes of reloading items of value, such as for resetting the number of uses permitted by a software license. The cited passages have nothing to do with a message schema defining messages to be sent to and received from a service; wherein the messages are defined in a data representation language, and wherein the data representation language is eXtensible Markup Language. In fact, Stern does not teach, in the Examiner's cited passage or elsewhere, anything regarding messages defined in a data representation language and wherein the data representation language is eXtensible Markup Language.

The combination of Juster in view of Stern, as suggested by the Examiner, fails to teach wherein the data representation language is eXtensible Markup Language. Additionally, the arguments presented above regarding claim 3 apply to claim 4, as well. Thus, for at least the reasons above, the rejection of claim 4 is not supported by the prior art and its removal is respectfully requested. Similar remarks as discussed above in regard to claim 4, apply to claim 17.

In regard to claim 5, contrary to the Examiner assertion, Juster in view of Stern fails to teach receiving an authentication credential indicating authorization to access the service; and wherein said linking comprises linking the authentication credential to the pre-generated message interface, wherein the message endpoint is configured to include

the authentication credential with each message to the address for a service in a distributed computing environment.

The Examiner cites portions of both Juster and Stern, without providing any reasons or discussion regarding how the Examiner is combining the teachings of Juster and Stern. The Examiner cites the Abstract, Figures 2A and 2B, and column 2, lines 4-26 of Juster and the Abstract, Figures 6 and 7, column 9, lines 3-25, column 12, lines 13-49 and column 13, lines 25-42 of Stern. However, none of the Examiner's cited portions of either Juster or Stern describes receiving an authentication credential indicated authorization to assess the service, linking the authentication credential to the pre-generated message interface, wherein the message endpoint is configured to include the authentication credential with each message sent to the address.

Instead, the cited portions of Juster teach that a client queries the server via the server's address request RPC endpoint to obtain the address of a RPC endpoint that provides a desired service (Juster, column 2, lines 18-23, column 4, lines 55-52, and column 7, lines 10-31). Once the client obtains the address, Juster only states that the client performs remote procedure calls on the server RPC endpoint using the received address (Juster, Figure 5, step 525, column 2, lines 23-25, and column 4, lines 62-64). The cited portions of Stern only refer to Stern's network interface device that stored identification keys for remote devices and object of value for network applications and that verify that a host computer is authorized to execute certain controlled applications.

The cited portions of Stern also fail to teach anything regarding linking an authentication credential to a pre-generated message, wherein the message endpoint is configured to include the authentication credential with each message sent to the address for a service in a distributed computing environment. Specifically, the Abstract and figures 6 and 7 of Stern provide an overview of Stern's secure, trusted network management function embedded within a Java enabled network interface device. Column 9, lines 3-25 describe how Stern's system allows for a Java Virtual Machine to provide secure storage of an object of value, such as a software license or other software

key controlling the use of an application and how a Java Virtual Machine may be treated as a trusted device for the purposes of reloading items of value, such as for resetting the number of uses permitted by a software license. Column 12, lines 13-49 of Stern describe how Stern's Java Enabled Network Interface Device may securely store digital signature objects while preventing unauthorized access or tampering of the stored secured objects. Additionally, column 13, lines 25-42 of Stern describe how Stern's system includes the capability of storing cryptographic keys for multiple users of a host computer allowing those users to "authenticate themselves without requiring additional hardware" (Stern, column 13, lines 30-34).

Thus, neither Juster nor Stern, singly or in combination, at the Examiner's cited portions or elsewhere, teach anything to do with linking an authentication credential, indicating authorization to access a service, to a pre-generated message interface, wherein the message endpoint is configured to include the authentication credential with each message sent to the address for a service in a distributed computing environment, as asserted by the Examiner.

Therefore, in light of the above remarks, applicants assert that the rejection of claim 5 is not supported by the cited art and withdrawal of the rejection is respectfully requested. Similar remarks as discussed above in regard to claim 5 apply to claim 18 as well.

Regarding claim 6, contrary to the Examiner's contention, Juster in view of Stern fails to teach locating a service advertisement for the service, wherein the service advertisement indicates an authentication service; requesting the authentication credential from the authentication service to assess the service; and wherein said receiving an authentication credential comprises receiving the authentication credential from the authentication service. The Examiner relies upon Stern and cites the Abstract, column 4, lines 1-14, column 9, lines 3-25, and column 10, lines 29-67. However, none of the cited passages of Stern teach anything regarding locating a service advertisement that indicates an authentication service, about requesting a authentication credential from the

authentication service to access the service, or about receiving an authentication credential from the authentication service. Instead, the Examiner has cited portions that provide an overview of Stern's secure, trusted network management function embedded within a Java enabled network interface device (Abstract, and column 4, lines 1-14). Column 9, lines 3-25 of Stern describe how Stern's system allows for a Java Virtual Machine to provide secure storage of an object of value, such as a software license or other software key controlling the use of an application and how a Java Virtual Machine may be treated as a trusted device for the purposes of reloading items of value, such as for resetting the number of uses permitted by a software license. Column 10, lines 29-67 describe how Stern's system allows for the storing of data, such as security objects, into flash memory, "so that they will remain in place after a power cycle or machine reboot."

Thus, the combination of Juster in view of Stern suggested by the Examiner fails to teach locating a service advertisement for a service, wherein the service advertisement indicated an authentication service; requesting an authentication credential from the authentication service to access the service; and receiving the authentication credential from the authentication service. For at least the reasons presented above, the rejection of claim 6 is not supported by the cited art and withdrawal of the rejection is respectfully requested. Similar remarks as discussed above in regard to claim 6 apply to claim 19 as well.

In regard to claim 7, Juster in view of Stern does not teach or suggest locating a service advertisement for a service, wherein the service advertisement comprises the address for the service and indicates a message schema for the service; wherein said receiving an address comprises receiving the address from the service advertisement; and wherein said linking comprises verifying that the pre-generated message interface corresponds to the message schema. The Examiner cites portions of both Juster and Stern, but none of the cited portions mentions anything regarding locating a service advertisement comprising an address for a service and that indicates a message schema for the service. The cited portions further fail to teach anything regarding receiving the

address for the service from the service advertisement or regarding verifying that a pre-generated message interface corresponds to the message schema.

Instead, the cited portions of Juster teach that a client queries the server via the server's address request RPC endpoint to obtain the address of a RPC endpoint that provides a desired service (Juster, column 2, lines 18-23, column 4, lines 55-52, and column 7, lines 10-31). Once the client obtains the address, Juster only states that the client performs remote procedure calls on the server RPC endpoint using the received address (Juster, Figure 5, step 525, column 2, lines 23-25, and column 4, lines 62-64). The Abstract and figures 6 and 7 of Stern provide an overview of Stern's secure, trusted network management function embedded within a Java enabled network interface device. Column 9, lines 3-25 describe how Stern's system allows for a Java Virtual Machine to provide secure storage of an object of value, such as a software license or other software key controlling the use of an application and how a Java Virtual Machine may be treated as a trusted device for the purposes of reloading items of value, such as for resetting the number of uses permitted by a software license. Column 12, lines 13-49 of Stern describe how Stern's Java Enabled Network Interface Device may securely store digital signature objects while preventing unauthorized access or tampering of the stored secured objects. Additionally, column 13, lines 25-42 of Stern describe how Stern's system includes the capability of storing cryptographic keys for multiple users of a host computer allowing those users to "authenticate themselves without requiring additional hardware" (Stern, column 13, lines 30-34).

Thus, the Examiner's combination of Juster in view of Stern fails to teach locating a service advertisement for a service, wherein the service advertisement comprises the address for the service and indicates a message schema for the service; wherein said receiving an address comprises receiving the address from the service advertisement; and wherein said linking comprises verifying that the pre-generated message interface corresponds to the message schema. Additionally, the arguments presented above regarding claim 1 also apply to claim 7.

Thus, for at least the reasons discussed above, the rejection of claim 7 is not supported by the prior art and removal thereof is respectfully requested. Similar remarks to those discussed above regarding claim 7 also apply to claim 20.

The Office Action rejected claims 8-14, 21-26 and 28 under 35 U.S.C. § 103(a) as being unpatentable over Hind et al. (U.S. Patent 6,585,778) (hereinafter "Hind") in view of Lee et al. (U.S. Patent 6,336,137) (hereinafter "Lee").

The Examiner has rejected claim 14 under 35 U.S.C. § 103(a) as being unpatentable over Hind in view of Lee. However, the Examiner has failed to provide any arguments regarding this rejection of claim 14. Applicants therefore assume that the reject of claim 14 under § 103(a) as unpatentable over Hind in view of Lee is a typographical error.

Regarding claim 8, contrary to the Examiner's assertion Hind in view of Lee fails to disclose a method for accessing services, comprising: <u>receiving a schema defining messages for accessing the service</u>; <u>generating message endpoint code according to said schema</u>.

Hind teaches a system for data policy enforcement using style sheet processing wherein a Document Type Definitions (DTD) including stored policy enforcement objects is applied to an input document representing a response to a user request. In the Examiner' cited passages (Abstract, column 4, lines 16-32, lines 50-59, and column 7, lines 9-18) Hind describes how a DTD corresponding to the input document may include references to stored policy enforcement objects that each enforce a data policy for an element of the input document and that each referenced stored policy object is instantiated and applied to the input document to create an output document matching the enforced data policies (Hind, column 4, lines 16-32). Hind further teaches that executing selected ones of the instantiated policy enforcement objects may also involve considering a target context of a user and may also involve determining whether an output DTD element will be created for an element of the input document (Hind column 4, lines 50-

59). The last cited passage describes how Hind's invention may be implemented as a software program running on an intermediary of a network or as individual modules invoked upon request (Hind, column 7, lines 9-18).

None of the cited passages describe receiving a schema defining messages for accessing a service. In contrast, Hind teaches a system that enforces data policy on retrieved documents (resulting from user requests). Hind fails to teach anything regarding schemas that define messages for accessing services. Instead Hind deals with data documents that result from user requests.

Hind also fails, contrary to the Examiner's contention, to teach generating message endpoint code according to the schema. Hind teaches the loading and executing of data policy objects that are referenced in a DTD defining data policies to be enforced on an input document representing a response to a user request. Such data policy objects are not message endpoint code, but instead are style sheets that translate, transform, or tailor the information of input documents before they are delivered to particular devices (Hind, column 7, lines 29-49, column 8, lines 38-57). Style sheet processing and data transformation is not the generation of message endpoint code according to a schema defining messages for accessing a service.

The Examiner admits that Hind does not teach linking said message endpoint code into executable operating code for the device and loading the message endpoint code and operating code onto the device and contends that Lee teaches such functionality. However, Lee discloses a method for transferring data via a network between a server and clients or browsers that are spatially distributed wherein a server parses client requests to determine both the language of the request and the information requested. The server also associates various markup languages with different virtual directories and clients that want to request a particular markup language can route their requests to the appropriate directory. (Lee, Abstract, column 4, lines 13-34). Lee fails to teach linking generated message endpoint code into executable operating code for the device and loading the message endpoint code and operating code onto the device.

The Examiner cites two passages of Lee in support of his arguments. The first passage describes how a client may be an HTTP client and also a gateway server to wireless browser clients that request pages from the client via WAP protocol that the gateway server transforms into HTTP/WML requests that are submitted to the main server. Such a gateway server also transforms the responses back into WAP/WML responses to be returned to the wireless clients (Lee, column 9, lines 21-45). The second passage cited by the Examiner describes a web engine that interprets a WML template containing embedded tags regarding what data the engine should retrieve from an associated database. The web engine then generates new WML code segments with the requested data and replaces the tags in the original WML template with the new code and sends the completed WML file to a WML browser (Lee, column 12, lines 22-38). Applicants can find no relevance in the Examiner's cited passages to linking generated message endpoint code into executable operating code for a device and loading the massage endpoint code and operating code onto the device. In contrast, Lee is teaching the use of specialized SWE tags for building custom data responses in a client requested mark up language. Lee *fails to teach* anything regarding any type of linking or loading of message endpoint code.

Hind and Lee, separately and in combination, fail to teach or suggest receiving a schema defining messages for accessing a service; generating message endpoint code according to the schema; and linking the message endpoint code into executable operating code for the device and loading the message endpoint code and operating code onto the device.

In the Response to Arguments section of the current Office Action, the Examiner asserts that Applicants' previous arguments do not clearly point out the patentable novelty that Applicants' claims present over the prior art. However, the statute clearly places a burden of proof on the Examiner which requires him to produce the factual basis for his rejection of an application under sections 102 and 103. *In re Warner*, 154 USPQ 173, 177 (C.C.P.A. 1967), *cert. denied*, 389 U.S. 1057 (1968). The Examiner has not met

this burden and Applicants' arguments, presented previously and discussed herein, clearly illustrate how the prior art, as cited by the Examiner, fails to disclose a method for accessing services, comprising: receiving a schema defining messages for accessing the service; generating message endpoint code according to said schema, as asserted by the Examiner. Furthermore, the Examiner failed to provide any specific response to Applicants' specific arguments when previously presented.

Therefore, in light of the above remarks, applicants assert that the rejection of claim 8 is not supported by the cited art and withdrawal of the rejection is respectfully requested. Similar remarks as discussed above in regard to claim 8 apply to claims 21 and 28.

Regarding claim 9, contrary to the Examiner's contention, Hind in view of Lee fails to disclose a method wherein said message endpoint is configured to <u>verify that the messages sent from the device to the service comply with the schema</u>. The Examiner cited two passages of Lee. The first (Lee, column 5, lines 26-50) describes Lee's client-server method where the server is configured to receive requests from and send responses to the client. The server is also configured to determine the language, protocol or syntax in the client requests and to interpret the request to determine the data submitted by the client. The system then recovers metadata or descriptive information from a metadata repository and creates a response in the client's preferred language, protocol, and/or syntax. Lee's system also includes the capability for interpreting the client request to determine classes or instances of business objects and user data to associate with the request and to determine data submitted by the client regarding creating, modifying, deleting, or appending such business objects. The second cited passage (Lee, column 7, lines 10-24) describes how, under Lee, the metadata may have different representations according to the client's preferred language, protocol, or syntax and how the server is configured to represent such metadata using the client's preferences.

Neither of the Examiner's cited passages have any relevance to a message endpoint configured to verify that messages sent from a device to a service comply with a

schema defining messages for accessing the service. In contrast, Lee teaches that a server determines a client's preferred language, protocol, and/or syntax and ensures that responses are appropriate for the client according to the client's preferred language, protocol, or syntax. Applicants also fail to see how Lee's title, "Web Client-Server System and Method for Incompatible Page Markup and Presentation Languages," as cited by the Examiner, is relevant to the Examiner's argument. None of these teachings in Lee have anything to do with a message endpoint being configured to verify that messages sent from the device to the service comply with a schema.

Thus, the combination of Hind in view of Lee, as suggested by the Examiner, fails to teach wherein the message endpoint is configured to verify that messages sent from the device to the service comply with the schema. Furthermore, the Examiner has failed to provide any specific response to applicants' arguments when previously presented.

Therefore, in light of the above remarks, applicants assert that the rejection of claim 9 is not supported by the cited art and withdrawal of the rejection is respectfully requested. Similar remarks as discussed above in regard to claim 9 apply to claim 22.

Regarding claim 10, Hind in view of Lee fails to teach a method wherein said <u>schema defines messages to be sent to and received from the service</u> wherein said <u>messages are defined in a data representation language</u>. In contrast, Hind teaches that intermediaries in his system apply various types of translation and/or transformations based upon target (client) context, giving the transformation of a response from XML to another data markup language as an example (Hind, column 7, lines 19-33). Hind also teaches how a DTD, as a definition of an XML document, tell a parser how to interpret the XML document, such as by defining entries for title, author, retail price, cost and quantity of a book, in one example (Hind, column 9, lines 27-35). Hind is describing various <u>data translations</u> and transformations on response <u>documents</u> in order to enforce specific data policies. Neither Hind nor Lee, separately or in combination, describes a schema that defines message in a data representation to be sent to and received from a

service. Furthermore, the Examiner has failed to provide any specific response to applicants' arguments when previously presented.

In light of the above remarks, the rejection of claim 10 is not supported by the cited art and withdrawal of the rejection is respectfully requested. Similar remarks as discussed above in regard to claim 10 apply to claim 23.

Applicants also assert that numerous other ones of the dependent claims recite further distinctions over the cited art. However, since the independent claims have been shown to be patentably distinct, a further discussion of the dependent claims is not necessary at this time.

CONCLUSION

Applicants submit the application is in condition for allowance, and notice to that effect is respectfully requested.

If any extension of time (under 37 C.F.R. § 1.136) is necessary to prevent the above referenced application from becoming abandoned, Applicants hereby petition for such extension. If any fees are due, the Commissioner is authorized to charge said fees to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/5181-66200/RCK.

Also enclosed herewith are the following items:

☒ Return Receipt Postcard

☐ Petition for Extension of Time

☐ Notice of Change of Address

☐ Fee Authorization Form authorizing a deposit account debit in the amount of $ for fees ( ).

☒ Terminal Disclaimer

Respectfully submitted,

Robert C. Kowert
Reg. No. 39,255
ATTORNEY FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8850

Date: January 4, 2005